IFW

PTO/SB/21 (08-00)
Approved for use through 10/31/02. OMB 0651-0031
Patent and Trademark Office: U.S. DEPARTMENT OF COMMENRCE
Under the Paper Reduction Act of 1995, no persons are required to respond to a collection of information unless it displays avalid OMB control number.

TRANSMITTAL FORM

(to be used for all correspondence after initial filing)

Application Number	10/609,208
Filing Date	June 27, 2003
First Named Inventor	Didier Poirot, et al.
Group Art Unit	2143
Examiner Name	NYA
Total Number of Pages in This Submission: 38	Attorney Docket Number: 15437-0637

ENCLOSURES ***(check all that apply)***

- ☐ Fee Transmittal Form
  - ☐ Fee Attached
- ☐ Amendment / Response
  - ☐ After Final
  - ☐ Affidavits/declaration(s)
- ☐ Extension of Time Request
- ☐ Express Abandonment Request
- ☐ Information Disclosure Statement
- ☒ Certified Copy of Priority Document(s)
  from France dtd 27 Mars 2003
- ☐ Response to Missing Parts/ Incomplete Application
  - ☐ Response to Missing Parts under 37 CFR 1.52 or 1.53
- ☐ Assignment Papers *(for an Application)*
- ☐ Drawing(s)
- ☐ Licensing-related Papers
- ☐ Petition
- ☐ Petition To Convert To a Provisional Application
- ☐ Power of Attorney, Revocation Change of Correspondence Address
- ☐ Terminal Disclaimer
- ☐ Request for Refund
- ☐ CD, number of CD(s)_____
- ☐ After Allowance Communication to Group
- ☐ Appeal Communication to Board of Appeals and Interferences
- ☐ Appeal Communication to Group *(Appeal Notice, Brief, Reply Brief)*
- ☐ Proprietary Information
- ☐ Status Letter
- ☐ Other Enclosure(s) *(please identify below):*

Remarks

SIGNATURE OF APPLICANT, ATTORNEY, OR AGENT

Firm or Individual name	Hickman Palermo Truong & Becker LLP Craig G. Holmes
Signature	
Date	June 3, 2004

CERTIFICATE OF MAILING

I hereby certify that this correspondence is being deposited with the United States Postal Service with sufficient postage as first class: mail in an envelope addressed to: Commissioner for Patents, P.O. Box 1450, Alexandria, VA 22313-1450 on this date: 6/3/04

Type or printed name	Annette Jacobs		
Signature		Date	June 3, 2004

Burden Hour Statement: This form is estimated to take 0.2 hours to complete. Time will vary depending upon the needs of the individual case. Any comments on the amount of time your are required to complete this form should be sent to the Chief Information Officer, Patent and Trademark Office, Alexandria, VA 22313-1450. DO NOT SEND FEES OR COMPLETED FORMS TO THIS ADDRESS. SEND TO: Commissioner for Patents, P.O. Box 1450, Alexandria, VA 22313-1450.

P7977

R E P U B L I Q U E F R A N Ç A I S E

02 p 80 29
Lettes (1)

INPI
INSTITUT NATIONAL DE LA PROPRIETE INDUSTRIELLE

BREVET D'INVENTION

CERTIFICAT D'UTILITÉ - CERTIFICAT D'ADDITION

COPIE OFFICIELLE

Le Directeur général de l'Institut national de la propriété industrielle certifie que le document ci-annexé est la copie certifiée conforme d'une demande de titre de propriété industrielle déposée à l'Institut.

Fait à Paris, le 27 MARS 2003

Pour le Directeur général de l'Institut
national de la propriété industrielle
Le Chef du Département des brevets

M Hauc

Martine PLANCHE

INSTITUT NATIONAL DE LA PROPRIETE INDUSTRIELLE
SIEGE
26 bis, rue de Saint Petersbourg
75800 PARIS cedex 08
Téléphone : 33 (0)1 53 04 53 04
Télécopie : 33 (0)1 53 04 45 23
www.inpi.fr

DB 267/141102
ETABLISSEMENT PUBLIC NATIONAL CREE PAR LA LOI N° 51-444 DU 19 AVRIL 1951

1er dépôt

INPI
INSTITUT NATIONAL DE LA PROPRIÉTÉ INDUSTRIELLE
26 bis, rue de Saint Pétersbourg
75800 Paris Cedex 08
Téléphone : 01 53 04 53 04 Télécopie : 01 42 94 86 54

BREVET D'INVENTION
CERTIFICAT D'UTILITÉ

cerfa N° 11354*01

Code de la propriété intellectuelle - Livre VI

REQUÊTE EN DÉLIVRANCE 1/2

Important ! Remplir impérativement la 2ème page.

Cet imprimé est à remplir lisiblement à l'encre noire DB 540 W /190600

Réservé à l'INPI

REMISE DES PIÈCES
DATE 28 JUIN 2002
LIEU 75 INPI PARIS
N° D'ENREGISTREMENT NATIONAL ATTRIBUÉ PAR L'INPI 0208079
DATE DE DÉPÔT ATTRIBUÉE PAR L'INPI 28 JUIN 2002

Vos références pour ce dossier *(facultatif)* SUN Aff. 42 (120748)

1 NOM ET ADRESSE DU DEMANDEUR OU DU MANDATAIRE À QUI LA CORRESPONDANCE DOIT ÊTRE ADRESSÉE

CABINET NETTER
36 avenue Hoche
75008 PARIS

Confirmation d'un dépôt par télécopie ☐ N° attribué par l'INPI à la télécopie

2 NATURE DE LA DEMANDE	Cochez l'une des 4 cases suivantes
Demande de brevet	☒
Demande de certificat d'utilité	☐
Demande divisionnaire	☐
Demande de brevet initiale	N° Date / /
ou demande de certificat d'utilité initiale	N° Date / /
Transformation d'une demande de brevet européen *Demande de brevet initiale*	☐ N° Date / /

3 **TITRE DE L'INVENTION** (200 caractères ou espaces maximum)

Localization Management server in a distributed computer system.

4 DÉCLARATION DE PRIORITÉ OU REQUÊTE DU BÉNÉFICE DE LA DATE DE DÉPÔT D'UNE DEMANDE ANTÉRIEURE FRANÇAISE	
	Pays ou organisation Date / / N°
	Pays ou organisation Date / / N°
	Pays ou organisation Date / / N°
	☐ **S'il y a d'autres priorités, cochez la case et utilisez l'imprimé «Suite»**

5 DEMANDEUR		☐ **S'il y a d'autres demandeurs, cochez la case et utilisez l'imprimé «Suite»**
Nom ou dénomination sociale		SUN MICROSYSTEMS, INC
Prénoms		
Forme juridique		
N° SIREN		
Code APE-NAF		
Adresse	Rue	901 San Antonio Road
	Code postal et ville	94303 PALO ALTO Californie
Pays		Etats-Unis d'Amérique
Nationalité		Société des Etats-Unis d'Amérique
N° de téléphone *(facultatif)*		
N° de télécopie *(facultatif)*		
Adresse électronique *(facultatif)*		

INPI
INSTITUT NATIONAL DE LA PROPRIETE INDUSTRIELLE

BREVET D'INVENTION
CERTIFICAT D'UTILITÉ

REQUÊTE EN DÉLIVRANCE 2/2

Réservé à l'INPI

REMISE DES PIÈCES
DATE 28 JUIN 2002
LIEU 75 INPI PARIS
N° D'ENREGISTREMENT NATIONAL ATTRIBUÉ PAR L'INPI 0208079

DB 540 W /190600

Vos références pour ce dossier : *(facultatif)*		SUN Aff. 42 (120748)
6 MANDATAIRE		
Nom		BEZAULT
Prénom		Jean
Cabinet ou Société		Cabinet NETTER
N °de pouvoir permanent et/ou de lien contractuel		
Adresse	Rue	36 avenue Hoche
	Code postal et ville	75008 PARIS
N° de téléphone *(facultatif)*		01 58 36 44 22
N° de télécopie *(facultatif)*		01 42 25 00 45
Adresse électronique *(facultatif)*		
7 INVENTEUR (S)		
Les inventeurs sont les demandeurs		☐ Oui ☒ Non **Dans ce cas fournir une désignation d'inventeur(s) séparée**
8 RAPPORT DE RECHERCHE		**Uniquement pour une demande de brevet (y compris division et transformation)**
Établissement immédiat ou établissement différé		☐ ☒
Paiement échelonné de la redevance		**Paiement en deux versements, uniquement pour les personnes physiques** ☐ Oui ☐ Non
RÉDUCTION DU TAUX DES REDEVANCES		**Uniquement pour les personnes physiques** ☐ Requise pour la première fois pour cette invention *(joindre un avis de non-imposition)* ☐ Requise antérieurement à ce dépôt *(joindre une copie de la décision d'admission pour cette invention ou indiquer sa référence)* :
Si vous avez utilisé l'imprimé «Suite», indiquez le nombre de pages jointes		

10 SIGNATURE DU DEMANDEUR OU DU MANDATAIRE (Nom et qualité du signataire)

N° Conseil 92-1024 (B) (M)
Jean BEZAULT

VISA DE LA PRÉFECTURE OU DE L'INPI

M. ROCHET

La loi n°78-17 du 6 janvier 1978 relative à l'informatique, aux fichiers et aux libertés s'applique aux réponses faites à ce formulaire. Elle garantit un droit d'accès et de rectification pour les données vous concernant auprès de l'INPI.

Localization Management server in a distributed computer system

The invention relates to a distributed computer system comprising computers or other hardware entities called nodes.

A distributed computer system comprises nodes associated to boards connected between them with e.g. ethernet links using ethernet switches and/or cPCI buses. In a distributed computer system, it is highly important to assign a specific role to a given node, e.g. to assign specific configuration parameters. For example, a node 20 may be assigned to run an application A on a "SPARC/Solaris" (SPARC is a Trademark of SPARC International Inc) board and a node 21 may be assigned to run another application B on a "Intel/Linux" board. To achieve this goal, a node localization is involved. Other improvements of node management may be performed when a node localization is provided.

A general aim of the present invention is to provide advances in this matter.

The invention concerns a management server of a distributed node system comprising :
- a table manager adapted to load a first table comprising for each of a plurality of node identifiers: a localization identifier, a hardware identifier, and configuration parameters,
- a node localization manager adapted to detect a new hardware identifier for a localization identifier and to send a modification message to the table manager, said modification message comprising a new hardware identifier for a localization identifier, the table manager being adapted to update such a first table responsive to the modification message,
- a client manager adapted to generate at least a second table in a client server according to the first table and to update said second table when the first table is updated.

The invention also concerns a method of node localization management in a distributed node system comprising the following steps:
a- loading a first table comprising for each of a plurality of node identifiers: a localization identifier, a hardware identifier, and configuration parameters,
b- detecting a new hardware identifier for a localization identifier

c- updating the first table responsive to the new hardware identifier for a localization identifier,

d- generating at least a second table in a client server according to the first table and updating said second table when the first table is updated.

Other alternative features and advantages of the invention will appear in the detailed description below and in the appended drawings, in which :

- figure 1 represents a distributed computer system comprising nodes and switches;

- figure 2 is a functional diagram of a distributed computer system using a network protocol module;

- figure 3 is a table of IP addresses stored according to the network protocol module;

- figure 4 is a functional diagram of a distributed computer system using a server according to the invention managing a network protocol module and switches;

- figure 5 is a functional diagram of the server according to the invention;

- figure 6 is a table of node localization generated by the server according to the invention;

- figure 7 is a flow-chart representing the initialization operations of the server according to the invention;

- figure 8 is a flow-chart representing the running operations of the server according to the invention;

- figure 9 is a flow-chart representing the method to generate a local table in case of a dynamic node localization and the use of an manageable switch;

- figure 10 is a flow-chart illustrating the method to update the local table in order to update the table of figure 6.

Additionally, the detailed description is supplemented with the following Exhibits:

- Exhibits A-1 and A-2 are examples of DHCP containers as described hereinafter,
- Exhibit B1 is an example of a node localization table according to the invention,
- Exhibits C-1 and C-2 are examples of functions used by the server of the invention.

These Exhibits are placed apart for the purpose of clarifying the detailed description, and of enabling easier reference. They nevertheless form an integral part of the description embodiments of the present invention. This applies to the drawings as well.

This invention also encompasses embodiments in software code, especially when made available on any appropriate computer-readable medium. The expression "computer-readable medium" includes a storage medium such as magnetic or optic, as well as a transmission medium such as a digital or analog signal.

This invention may be implemented in a network comprising computer systems. The hardware of such computer systems is for example as shown in Fig. 1, where in the computer system N4:

- 1 is a processor, e.g. an Ultra-Sparc (SPARC is a Trademark of SPARC International Inc);
- 2 is a program memory, e.g. an EPROM for BIOS;
- 3 is a working memory for software, data and the like, e.g. a RAM of any suitable technology (SDRAM for example); and
- 7 is a network interface device connected to communication links L1 and L2, each in communication with a switch S1, S2 to enable communication with other computers. Network interface device 7 may be an Ethernet device, a serial line device, or an ATM device, inter alia. Links L1 and L2 may be based on wire cables, fiber optics, or radio-communications, for example.

The computer system, also called node Ni, may be a node amongst a group of nodes in a distributed computer system. Some nodes may further comprise a mass memory, e.g. one

or more hard disks. The nodes having hard disk are designated as diskfull and other nodes having no hard disk are designated as diskless.

Data may be exchanged between the components of Figure 1 through a bus system 9, schematically shown as a single bus for simplification of the drawing. As is known, bus systems may often include a processor bus, e.g. of the PCI type, connected via appropriate bridges to e.g. an ISA bus and/or an SCSI bus.

Node N4 is one of the nodes of the cluster NM, NVM, N3. For reliability reasons, the cluster may comprise a master node NM and a vice-master node NVM adapted to manage the nodes of the cluster. When informed about the master node failure, the vice-master node may replace the failed master node in its role.

References to the drawings in the following description will use two different indexes or suffixes i and j, each of which may take anyone of the values: {NM, NVM, 3...,n} n being the number of nodes in the cluster. In the foregoing description, a switch is only an example of a connection entity for nodes on the network, cPCI buses may also be used.

Each node Ni is connected to a network, e.g. the Ethernet network 31 which may be also the internet network. The node Ni is firstly connected to a switch S1, e.g. an Ethernet switch, capable of interconnecting the node Ni with other nodes Nj through the network 31. The switch comprises several ports P, each being capable of connecting a node Ni to the switch S1 via a link L1. In an embodiment of a switch, the number of ports per switch is limited, e.g. to 24 ports in some switch technologies. Several switches may be linked together in order to increase the number of nodes connected to the network, e.g. the ethernet network. By way of example only, the switch may be called an ethernet switch if the physical network is an ethernet network. Indeed, different switch types exist such as ethernet switch and internet switch also called IP switch. Each switch has an identifier :

- for an ethernet switch, the identifier is e.g. a MAC address being an ethernet address or an IP address for administration,
- for an IP switch, the identifier is e.g. an IP address.

Each switch port has an identifier, e.g. a port number being generally an integer or an ethernet port address.

In the following description, an ethernet switch is used but the invention is not restricted to this switch type.

If desired, for availability reasons, the network 31 may be also redundant, e.g. the ethernet network. Thus, the links L1 may be redundant: nodes Ni of the cluster are connected to a second network 32 via links L2 using a redundant switch as a switch S2. This redundant network is adapted to interconnect a node Ni with another node Nj through the network 32. For example, if node Ni sends a packet to node Nj, the packet may be therefore duplicated to be sent on both networks. In fact, the second network for a node may be used in parallel with the first network or replace it in case of first network failure.

Also, as an example, it is assumed that packets are generally built throughout the network in accordance with a transport protocol and a presentation protocol, e.g. the Ethernet Protocol and the Internet Protocol. Corresponding IP addresses are converted into Ethernet addresses on Ethernet network.

A boot of a node is hereinafter describes.

When a new node is inserted in the system at initialization time or at any other time, the node is booted according to its software load configuration parameters. Particularly, this boot enables a diskless node to obtain its link addresses from a diskfull node.

At initialization time, each node executes an initialization software providing various capabilities such as low level hardware configuration, low level hardware tests, operating system loading configuration, boot file configuration. On Sun hardware, such software may be called Open Boot Prom (OBP).

According to the configuration of this initialization software (e.g. OBP), it launches a program to allow diskfull nodes to boot from their local disk or from the network, and to allow diskless nodes to boot from the network only. To boot from the network means to boot from the disk of a remote diskfull node in the network.

α

In the context of this invention, boot of diskless nodes is particularly considered. When a diskless node boots, its initialization software (e.g. OBP) sends a broadcast request (e.g. DHCP discover) containing an identifier of the hardware of the node, e.g. its board ethernet MAC address, a board serial number, according to a particular protocol, e.g. DHCP. Thus, all nodes may receive this request. As illustrating in figure 2, a diskfull node comprises a DHCP server 106 adapted to reply to this request by providing

- its address as the DHCP server,
- the file path name of the diskfull node to download the default boot file on the diskless node.

This DHCP server replies also by providing an address, e.g. IP address, becoming the IP address of the diskless node. Each data or resources may be contained in a disk 4, more precisely in portions of disk called DHCP containers 41 and 42.

Several diskfull nodes have a DHCP server adapted to reply to a request of a diskless node, called a client's request. A function exported by a public module of a service provider layer of DHCP server allows two DHCP servers to run on two separated diskfull nodes and sharing the same DHCP containers 41 and 42 on the disk 4-NM. The second disk 4-NVM also comprises mirrored DHCP containers 41 and 42 being used in case, for example, of master node failure.

Figure 2 illustrates the anterior art of the management of a DHCP server.

Thus, each diskfull node comprises a DHCP function 106, also called the DHCP server 106, composed of a core DHCP server, a configuration file tool and a public module e.g. NHRBS public module (Sun Netra Highly Reliable Boot Service).

The configuration file tool is adapted to configure the core DHCP server so as to use the public module. The public module may be a dynamic library automatically loaded at run-time by the core DHCP server.

As seen, the DHCP server 106 is linked to a disk 4 having containers e.g. containers 41, 42, via a NFS server 104 (Network file system). The configuration file tool indicates the link

level interfaces of the diskfull node to which the DHCP server is connected to and that this DHCP server monitors. Thus, one container is designated for each link level interface monitored by the DHCP server. This container is called the *network* container 41. It may contain the data corresponding to request received at the link level interface of the diskfull node. Its address may indicate this link level interface address. Indeed, there may be one network container per sub-net managed by the DHCP server. Typically, a single server running on a system equipped with two network interfaces (hme0 and hme1 for example on the Netra systems), can manage two DHCP network containers, one for each interface. In the example of exhibit A-1, the network containers used by the SUNnhrbs module are named: *SUNWnhrbsN_A_B_C_D* where:

. *N* is the version of the public module managing the container. (e.g. 1 for NHAS 2.0).

. *A_B_C_D* is the classical ASCII decimal representation for the sub-net corresponding to the network interface,

For example, if interface hme0 is connected to sub-net 10.1.1.0 and hme1 to sub-net 10.1.2.0, the network containers will be named: *SUNWnhrbs1_10_1_1_0* and *SUNWnhrbs1_10_1_2_0*. The content of the network containers used by SUNWnhrbs may be compatible with the ones used by the public module of the service provider layer.

These network containers 41 are used by the public module to store data associated in a table T as illustrated in figure 3.

The network container 41 may contain entries having different fields such as:

- an C-IP field containing the IP address managed by the DHCP server,

- an H-ID field containing, when the IP address is assigned to a client node, the identifier of the hardware associated to said node, e.g. the MAC address of said node,

- an S-IP field containing the IP address of the DHCP server owning this entry, e.g. the server which manages it.

Other fields may be added to specify other data related to the entry.

α

The content of these network containers 41 may be rendered compatible with the containers used for other modules. In the anterior art, the *dhcp network* container 41 and the *dhcptab* container 42 are configured by the configuration file tool. Moreover, in the anterior art, the core DHCP server is composed of at least an application layer comprising applications to update DHCP containers.

An example of *dhcp network* container configured by the configuration file tool is given in Exhibit A-1 for the sub-network `10_1_1_0`. The entry concerning the IP address 10.1.1.10 for a node which has an hardware identifier which is 01080020F996DA. An example of *dhcptab* container is given in Exhibit A-2. This *dhcptab* container contains definition for configuration parameters which can be applied to one or more network container entries.

The containers of diskfull nodes may be shared by at least two diskfull nodes, the master node and the vice-master node. The diskfull nodes may use a NFS network (Network File System).

Amongst diskfull nodes, the master NM and vice-master NVM nodes of the cluster are to be designated initially and at every boot of the system. Moreover, in the anterior art, the DHCP server is linked to the management layer of the node. Thus, when the management layer 11-NM detects the master node NM as failed, the vice-master NVM is designated as the new master node of the cluster. The management layer 11-NVM detects the vice-master node as the current master node and the DHCP server of the node is informed.

The DHCP server is adapted to assign to a booting node providing its board hardware identifier, an available IP address indicating configuration parameters. This assignment is random. If the node fails and re-boots, the DHCP server may assign to this node another available IP address. Thus, if a node re-boots, the IP address of this node may change, causing the change of configuration parameters which provokes compatibility problems between the board type, the operating system and the applications running on the board. Moreover, the board of the node may be changed providing a new board hardware identifier. A requirement is to provide personalized configuration parameters for a node, even in case of node re-boot or board change.

To solve these problems, the invention proposes a node localization server to manage services requiring node localization as the DHCP server.

The hereinafter description refers to figures 4 and 5.

Figure 4 illustrates a cluster supporting redundancy of master node for reliability only in an example. The diskfull nodes of the cluster being the master and the vice-master nodes NM and NVM comprise a management layer 11 connected to a node localization server 22. This management layer is adapted to notify the node localization server (NLS) 22 about the elected master node or the master node failure. The node localization server (NLS) 22 comprises a NLS core 223 being a table manager, a NLS client 221 and a node localization manager 222, e.g. a SNMP manager.

The NLS core 223 manages a table, the Node Localization Table (NLT), containing information for each node's geographical location, e.g. Ethernet switch port number and Ethernet board MAC addresses. This table may be cached in memory 4, e.g. a cache memory, more particularly in a portion 40 of this cache memory, and may reside in a mirrored file system for redundancy acceded only through a NFS server 27 even locally.

The NLS client 221 comprises a client manager 231 connected to the services requiring node localization as the DHCP server 24. As described hereinafter, the node localization server 22 is adapted to start at initialization time before the DHCP server. The node localization server 22 is adapted to configure and to load the node localization table NLT as illustrated in figure 6 and to configure the DHCP containers 41 and 42 according to this table NLT using configuration file tool 43 of the DHCP server. This NLT table may be managed dynamically or statically as seen hereinafter. Thus, the DHCP containers 41 and 42 are updated when changes appear in the NLT table.

In case of a dynamic table management, the node localization server 22 establishes an optional node localization manager 222 using the NLS localization module API 240 and *nlsInitialize()* function in Exhibit C1. The node localization manager 222 comprises a localization dynamic module being e.g. an Ethernet switch SNMP manager 241 adapted to:

- use a network information management protocol, e.g. the simple network management protocol (SNMP) as described in RFC 1157 (may 1990), in order to establish a connection with and to work in relation with an agent module, e.g. an agent module 26-S1 of a switch S1, using advantageously the same network information management protocol (SNMP),
- request the agent module to perform network management functions defined by the SNMP protocol, such as a *get-request(var)* function requesting the agent module to return the value of the requested variable *var*, a *get-next-request(var)* function requesting the agent module to return the next value associated with a variable e.g. a table that contains a list of elements, the *set-request(var,val)* function requesting the agent module to set the value *val* of the requested variable *var*.

The goal of this module is to detect the active ports of switches (traffic detected) and, for each active port having a given port number, to retrieve the hardware identifier of the connected node. Thus, the Ethernet switch SNMP manager 241 develops a method to retrieve node localization at runtime described in figure 9 and 10. The Ethernet switch SNMP manager 241 is based on both *RFC1213* (Management Information Base (MIB-II) for use with network management protocols in TCP/IP- based internets) and *RFC1493* (portion of the Management Information Base (MIB) for use with network management protocols in TCP/IP based internets). The NLS localization module API 240 is used by NLS core 223 to manage the node localization manager 222. Through this NLS localization module API 240 comprising functions described in Exhibit C1 and implemented by the NLS localization module, the NLS core is able to

*read the private configuration (*nlsConfigure*) of the localization dynamic module, e.g. Ethernet switch SNMP manager 241, and initialize by essentially creating an SNMP session (*nlsInitialize*)

* to register with the localization dynamic module to receive localization information changes, e.g. to register a callback function to be invoke on SNMP dynamic information messages called "traps" (*nlsRegister*),

* to make it operational by starting it (*nlsStart*),

* to get all necessary information (*nlsLocalize*) from the switches and to build an internal localization table for each switch,

* finally to stop and terminate properly the Node localization manager 222.

α

Using a *nlsProcessLocalize()* function described in Exhibit C2, the NLS localization handler 232 notifies the NLS core 223 of node localization changes for the NLS core to update the dynamic table according to node localization information changes.

Figure 6 illustrates the NLT table providing the node's geographical localization. It may comprise the following fields :

- a node identifier, being a unique number to identify the node (N-ID),
- a localization type (Loc-T) which can have the following value :

  00: no localization is required for this node. In this case the localization Identifier and the hardware Identifier are not relevant.

  01: Static localization is required for this node. The localization Identifier is not relevant and the hardware Identifier is set at initial configuration time.

  02: Dynamic configuration is required for this node. The node localization manager is configured and the hardware Identifier field is initialized to 0.

- a Localization identifier (Loc-ID) : This identifier is localization type's dependent.

  Loc-T = 00 :This localization identifier is not relevant.

  Loc-T = 01 : for static localization, it may have the following format: *subnet. Subnet* is the subnet IP address configured on network interface of ethernet address <hard-wareId>.

  Loc-T = 02 : for dynamic policy, it may have the following meaning: *id@subnet. Id* being the port number and *subnet* being the subnet IP address managed by the ethernet switch for switch port-based localization.

- a hardware identifier (H-ID) : This identifier is localization type's dependent.

  Loc-T = 01 : for static localization, it may be set with the client identifier used by OBP in the DHCP discover request.

  Loc-T = 02 : for dynamic localization, the hardware identifier is set to 0. It is filled at node localization manager 222 startup time and updated at run-time if the node localization server 22 detects a modification.

- bootParameters (B-Par) : it represents the DHCP various parameters used at boot time. Pathname of root file system to mount, pathname of boot image file, pathname of swap file for example.

In an embodiment, the IP address of the subnet may have the following pattern : *10.<cluster ID>.<1/2>. 0*. The *<1/2>* term means the term may have for value 1 or 2 for a subnet or its redundant subnet. The IP address of a node may be deduced from the IP address of the subnet as it may have the following pattern : *10.<cluster ID>.<½>.<nodeID>*. When a DHCP container is configured according to the NLT table or updated, the IP addresses of nodes are also furnished from the NLT table.

Thus, the node being on a given port may always have the same IP address and the same configuration parameters even in case of node re-boot or change of board for a node.

A NLT table example is developed in Exhibit B1. This example illustrates 3 different cases. Node 20 is not bound to any hardware board and can be affected to one of the hardware board not localized. Node 21 is bound statically to the board having 080020f99914 MAC address. Node 22 is linked to the hardware board connected to port number 4 of each Ethernet switch. The MAC address used as client identifier by DHCP to boot the node is filled at node localization server 22 startup time as described in flow-chart of figure 7.

Flow-chart of figure 7 illustrates the start-time of the node localization server.

At initialization time, a NLS daemon in charge of "NLS availability" is started on both master and vice-master node. The two instances of NLS daemon open a persistent connection with the CMM to get the current and receive the future cluster member-ship information at operation 801. Then, NLS takes two different paths, depending on whether it find itself on the Master or on theVice-master. The node localization server is started after the management layer has elected a master node. At operation 802, if the node localization server is running on the master node, it offers the localization service. The other one running on the vice-master node is waiting for management layer notifications at operation 803. Entering in a stand by mode, the vice-master node is only waiting for cluster membership notification. When it receives a CMM_MASTER_ELECTED notification, it checks the role assigned to the node it is running on. If it runs on the master node at operation 802, it takes the primary role by starting the NLS service.

For the node localization server running on the master node, at startup time, the NLS daemon uses a general configuration file managed by NLS core, to get information about NLS service configuration at operation 804 and to instantiates and initializes NLS core by setting its configuration from this general configuration file at operation 806. Then, using the configuration information gathered previously, the Node Localization Table (NLT) is loaded in memory at operation 808. If a node localization manager has been configured, e.g. in the general configuration file, it is instantiated, initialized and configured with specific mandatory/optional module parameters at operation 810. Thus , it establishes a connection, an SNMP session, with a connection entity, e.g. an ethernet switch being an SNMP agent. The node localization manager callback function is registered in order to be ready to receive and process all the node localization information from , e.g. the SNMP agent. Then, NLS core gets, from the node localization manager, the node localization information and updates accordingly the NLT table in memory. From the NLT table now up to date, NLS core can creates/updates DHCP containers (*network* and *dhcptab*) using standard DHCP administration commands at operation 812. As described hereinabove, when a DHCP container is configured or updated according to the NLT table, the IP addresses of nodes are generated from the IP addresses of subnets furnished from the NLT table by the NLS core. These IP addresses of nodes are used for the DHCP containers configuration. The start-time ends.

Figure 8 illustrates the node localization server during run-time, following the start-time of the flow-chart of figure 7. Once the operations of figure 7 completed, NLS core is managing administration events coming from NLS availability daemon like "stop the NLS service" in case of master switch-over event. In this case at operation 902, the flow-chart continues at operation 803. NLS core is also managing administration events coming from NLS availability daemon like"hang-up NLS service" which means re-starting from a table NLT located on the mirrored disk, the NLT may have been modified by an administrator. NLS core starts also listening to localization events at operation 904 coming from node localization manager, a localization event informing about the new hardware identifier for a given localization identifier. In this case, the NLS core manages the update of the NLT table according to changes of hardware identifier for a given localization identifier (or a given node identifier). The NLS core is also adapted to update the DHCP containers according to the new NLT table. The hang-up event will lead to a complete re-generation of NLT table from persistent storage, that is from the disk.

Figure 9 and 10 illustrate the method to retrieve node localization of the Ethernet switch SNMP manager 241. In the following description, the "manager module" represents the Ethernet switch SNMP manager 241 of the master node.

In figure 9, the process is aimed to build a local table of at least data couples indicating port identifier/hardware identifier. In operation 601, the manager module requests an agent module of a switch designated with its identifier (e.g. IP address of the switch or the name of the switch) for the status of a given port designated with its port identifier (e.g. its port number). At operation 602, the agent module having retrieved this port status, sends the port status and other additional information to the manager module of the master node.

If the port status indicates that the port is up and that a node is connected to this port and its hardware identifier is known at operation 604 ("learned"), the manager module may request the agent module to determine this hardware identifier at operation 608. The agent module may retrieve this information in a Management Information Base implemented in the agent module and sends it to the manager module. At operation 610, the manager module retrieves the hardware identifier corresponding to the port number and stores the data couple in a table, this data couple indicating at least the hardware identifier and the port identifier. At operation 610, a data couple corresponding to the same port identifier may be already stored in the table. In this case, the retrieved hardware identifier (new node identifier) and the hardware identifier already stored in the table (old hardware identifier) are compared and responsive to a difference between them, the old hardware identifier is replaced by the new hardware identifier : the data couple in the table is thus updated. If other ports may be requested by the manager module at operation 612, the process returns to operation 601, else it ends.

If the port status indicates that the port is down, or if the port status indicates that a node is connected to this port and without indicating that the hardware identifier is known (or indicating that the hardware identifier is not known) at operation 604, the manager module may restart operations 601 to 604 for this port. The manager module restarts operations 601 to 604 for this port until the port status is *up* at operation 604 or until at operation 605 the manager module has restarted R consecutive times operations 601 to 604 for this port, R being an integer greater than 1. In this last case, the flow-chart continues at operation 612.

The flow chart of figure 10 may be repeated regularly to request for hardware identifier connected to a port identifier in order to update the table and to maintain a current table.

A manager module may execute the flow-chart of figure 9 in parallel for different ports in an agent module or in several agent modules, thus having an local table for each switch.

In figure 10, a modification of the status of a port may appear in the switch, that is to say, a node having a down status may change to an up status and reciprocally. In this case, an agent module sends a trap() function, as described hereinbefore, in operation 702. The manager module receives then this trap at operation 704. If the port status indicates the value *up*, at operation 710 the flow-chart continues in figure 9 operation 601 to build a new version of its local table. For an already stored data couple in the manager module's memory, the manager module retrieves the hardware identifier for the port and updates the already stored data couple in operation 610 of figure 9. If the port status indicates the value *down* at operation 706, the data couple in the manager module's memory is invalidated at operation 708. After operations 708 or 710, the flow-chart ends.

At operation 704, if some differences are found between the old and the new version of the local table, the node localization handler notifies the NLS core by calling the callback function previously registered, giving in argument a couple of information (locID, hardwareID), where locId may be portNumber@subnet and hardwareID may be the MAC address. The NLT table is thus updated.

The invention is not limited to the hereinabove embodiments. Thus, the table of the manager module's memory may comprise other columns or information concerning for example the time at which the information for the port and the connected node is retrieved. The manager module may regularly request for information such as the port status and the node identifier connected to this port. The manager module may define a period of time to retrieve said information. In an embodiment, the table may also indicate all the ports and their status. If the node has a down status or if it is not identified, the column C2 is empty. This enables the manager module, the node having the manager module, or a user requesting this node, to have a sort of map for ports of a switch and to know to which port the node is connected.

α

If the port of a node is down, this port status is indicated in the table and the node connected to this port may be changed and may be connected to another port having an *up* status.

The invention is not limited to the hereinabove features of the description.

The node localization manager 222 may comprise the node localization handler 232.

Though the description is based on the DHCP server, other services may be managed by the node localization server according to the invention. Moreover, the description is based on nodes interconnected with switches having ports. The invention is also applicable to nodes interconnected on cPCI buses having slots, the hardware identifier being for example a string.

The invention also covers a software product comprising the code for use in the manager server of the invention.

α

Exhibit A

A-1 Example of Dhcp network container

```
$ cat /SUNWcgha/remote/var/dhcp/SUNWnhrbs1_10_1_1_0
# SUNWnhrbs1_10_1_1_0#
# Do NOT edit this file by hand -- use pntadm(1M) or
dhcpmgr(1M) instead#
10.1.1.12|00|01|10.1.1.1|4294967295|2774498845436936197|pn1
2|netra-t1-9
10.1.1.11|00|01|10.1.1.1|4294967295|2774498845436936198|pn1
1|netra-t1-8
10.1.1.10|01080020F996DA|01|10.1.1.1|4294967295|13030884046
819819544|pn10|netra-t1-7
```

A-2 Example of Dhcptab container

$cat /SUNWcgha/remote/var/dhcp/SUNWnhrbs_dhcptab

```
# SUNWnhrbs1_dhcptab
#
# Do NOT edit this file by hand -- use dhtadm(1M) or
dhcpmgr(1M) instead
#
Locale|m|15358963579193655297|:\
:UTCoffst=-18000:\
:BootSrvA=10.1.1.1:\
:BootSrvN="cgtp-master-link-a":\
:Subnet=255.255.255.0:

pn10|m|2508223517468655617|\
:Include=Locale:\
:BootFile="inetboot.sun4u.Solaris_8":\
:SrootNM="cgtp-master-link-a":\
:SrootIP4=10.1.1.1:\
:SrootPTH="/export/home/client/netra-t1-7/root":\
```

```
:SswapPTH="/export/home/client/netra-t1-7/swap":\
:SswapIP4=10.1.1.1:\
:Broadcst=10.1.1.255:\
:Router=10.1.1.1:
#
pn11|m|3894769252745347073|:\
:Include=Locale:\
:BootFile="inetboot.sun4u.Solaris_8":\
:SrootNM="cgtp-master-link-a":\
:SrootIP4=10.1.1.1:\
:SrootPTH="/export/home/client/netra-t1-8/root":\
:SswapPTH="/export/home/client/netra-t1-8/swap":\
:SswapIP4=10.1.1.1:\
:Broadcst=10.1.1.255:\
:Router=10.1.1.1:
#
SbootFIL|s|11187222949365022721|Vendor=SUNW.UltraSPARC-IIi-
cEngine,7,ASCII,1,0
SswapPTH|s|15424547248767238145|Vendor=SUNW.UltraSPARC-IIi-
cEngine,6,ASCII,1,0
SswapIP4|s|6900077579085021185|Vendor=SUNW.UltraSPARC-IIi-
cEngine,5,IP,1,0
SrootPTH|s|5589811562496917505|Vendor=SUNW.UltraSPARC-IIi-
cEngine,4,ASCII,1,0
SrootNM|s|17526602374842417153|Vendor=SUNW.UltraSPARC-IIi-
cEngine,3,ASCII,1,0
SrootIP4|s|1126181381819334657|Vendor=SUNW.UltraSPARC-IIi-
cEngine,2,IP,1,1
SrootOpt|s|9453337092827381761|Vendor=SUNW.UltraSPARC-IIi-
cEngine,1,ASCII,1,0
```

α

Exhibit B

B-1 Node Localization Table

SUNWnhnlt

Node Localization Table

#

nodeId|locType|locId|hardwareId|BootParameters

#

20|00|00|00|BootFile=inetboot.sun4u.Solaris_8:SrootPTH=/export/root/NMEN-C11-N20:SswapPTH=/export/root/NMEN-C11-N20

21|01|10.11.1.0|080020f99914|BootFile=inetboot.sun4u.Solaris_8:SrootPTH=/export/root/NMEN-C11-N21:SswapPTH=/export/root/NMEN-C11-N21

21|01|10.11.2.0|080020f99915|BootFile=inetboot.sun4u.Solaris_8:SrootPTH=/export/root/NMEN-C11-N21:SswapPTH=/export/root/NMEN-C11-N21

22|02|04@10.11.1.0|0|BootFile=inetboot.sun4u.Solaris_8:SrootPTH=/export/root/NMEN-C11-N22:SswapPTH=/export/root/NMEN-C11-N22

22|02|04@10.11.2.0|0|BootFile=inetboot.sun4u.Solaris_8:SrootPTH=/export/root/NMEN-C11-N22:SswapPTH=/export/root/NMEN-C11-N22

Exhibit C

C-1 Functions of NLS localization module API:

Void nlsConfigure(String config) Pass a configuration string to the node localization manager

Void nlsFinalize() Terminate properly node localization manager

Void nlsInitialize() Initialization of node localization manager

Void nlsLocalize(NlsLocInfoList locInfoList) Collect a list of localization information represented by couples (loc-ID,H-Id).

Void nlsRegister(NlsLocalizationModuleHandler handler) Register a callback with the node localization manager

Void nlsStart() Start localization service.

Void nlsStop() Stop localization service.

C-2 Function of the NLS localization handler

Void nlsProcessLocalize(NlsLocInfo locinfo) Notify NLS core of a localization information change returning. (loc-ID, H-ID).

Claims

1. A management server of a distributed node system comprising :

- a table manager (223) adapted to load a first table (NLT) comprising for each of a plurality of node identifiers (N-ID): a localization identifier (Loc-ID), a hardware identifier (H-ID), and configuration parameters (B-Par),

- a node localization manager (222,232) adapted to detect a new hardware identifier (H-ID) for a localization identifier (Loc-ID) and to send a modification message to the table manager (223), said modification message comprising a new hardware identifier for a localization identifier, the table manager (223) being adapted to update the first table (NLT) responsive to a such modification message,

- a client manager (231) adapted to generate at least a second table (T) in a client server (24) according to the first table (NLT) and to update said second table (T) when the first table (NLT) is updated.

2. The management server of claim 1, wherein the node localization manager is operative in a dynamic node localization mode.

3. The management server of any of the preceding claims, wherein the node identifier (N-ID) comprises a number different for each node of the distributed node system.

4. The management server of any one of the preceding claims, wherein, in a static node localization mode, the localization identifier (Loc-ID) comprises an Internet Protocol address of the network.

5. The management server of any one of the preceding claims, wherein, in a dynamic node localization mode, the localization identifier (Loc-ID) comprises a port number (PN), a node being attached to the corresponding port, and an Internet Protocol address of the network.

d

6. The management server of any one of the preceding claims, wherein the hardware identifier (H-ID) comprises the ethernet address of the node in the distributed node system.

7. The management server of any one of the preceding claims, wherein, in the dynamic node localization mode and nodes being attached to ports of a switch in the distributed node system, the node localization manager (222,232) is adapted, at initialization time, to request for port status for each port number and to retrieve the port status indication.

8. The management server of any one of the preceding claims, wherein, in the dynamic node localization mode and nodes being attached to ports of a switch in the distributed node system, the node localization manager (222,232) is adapted to receive a message comprising a modified port status indication.

9. The management server of any one of the preceding claims, wherein, responsive to port status meeting a given condition, the node localization manager (222,232) is adapted to retrieve hardware identifier (H-ID) of the node connected to said port.

10. The management server of claim 8, wherein the given condition comprises the fact that the port status is up and indicates that the hardware identifier (H-ID) is known.

11. The management server of any one of the preceding claims, wherein the node localization manager (222,232) is a manager using the SNMP protocol adapted to establish a connection with an agent (26) of the switch using the same SNMP.

12. The management server of any one of the preceding claims, wherein the client server (24) comprises a server using the DHCP protocol.

13. The management server of any one of the preceding claims, wherein the client manager (231) is adapted to generate the Internet Protocol address of the nodes using the network

Internet Protocol address and to generate at least the second table (T) using the Internet Protocol address of the nodes, the corresponding hardware identifier (H-ID) of the first table (NLT) and the configuration parameters (B-Par).

14. The management server of claim 1, wherein the hardware identifier is a slot identifier configured at initialization time.

15. A method of node localization management in a distributed node system comprising the following steps:

a. loading a first table comprising for each of a plurality of node identifiers : a localization identifier, a hardware identifier, and configuration parameters (808),

b. detecting a new hardware identifier for a localization identifier (810, 904)

c. updating the first table responsive to the new hardware identifier for a localization identifier (810, 906),

d. generating at least a second table in a client server according to the first table and updating said second table when the first table is updated (812, 908).

16. The method of claim 15, wherein steps b. and c. are operative in a dynamic node localization mode.

17. The method of any one of claims 15 to 16, wherein the node identifier of step a. comprises a number different for each node of the distributed node system.

18. The method of any one of claims 15 to 17, wherein, in a static node localization mode, the localization identifier comprises an Internet Protocol address of the network.

19. The method of any one of claims 15 to 18, wherein, in a dynamic node localization mode, the localization identifier comprises a port number, a node being attached to-the corresponding port, and an Internet Protocol address of the network..

20. The method of any one of claims 15 to 19, wherein the hardware identifier comprises the ethernet address of the node in the distributed node system.

21. The method of any one of claims 15 to 20, wherein, in the dynamic node localization mode and nodes being attached to ports of a switch in the distributed node system, step b. comprises, at initialization time, requesting for port status for each port number (601) and retrieving the port status indication (604).

22. The method of any one of claims 15 to 21, wherein, in the dynamic node localization mode and nodes being attached to ports of a switch in the distributed node system, step b. comprises receiving a message comprising a modified port status indication (704, 706).

23. The method of any one of claims 15 to 22, wherein step b. comprises

b1. responsive to port status meeting a given condition, retrieving hardware identifier of the node connected to said port (608, 610).

24. The method of claim 23, wherein the given condition of step b1. comprises that the port status is up and indicates that the hardware identifier is known (604).

25. The method of any one of claims 15 to 24, wherein step b. comprises establishing a connection with an agent of the switch using the SNMP protocol (810).

26. The method of any one of the claims 15 to 25, wherein the client server of step d. comprises a server using the DHCP protocol.

27. The method of any one of claims 15 to 26, wherein step d. comprises generating the Internet Protocol address of the nodes using the network Internet Protocol address and generating at least the second table using the Internet Protocol address of the nodes, the corresponding hardware identifier of the first table and the configuration parameters (812).

28. The method of claim 15, wherein the hardware identifier is a slot identifier configured at initialization time.

29. A software product comprising the code for use in the manager server as claimed in any of claims 1 to 14.

2 (25 pages) CABINET NETTER

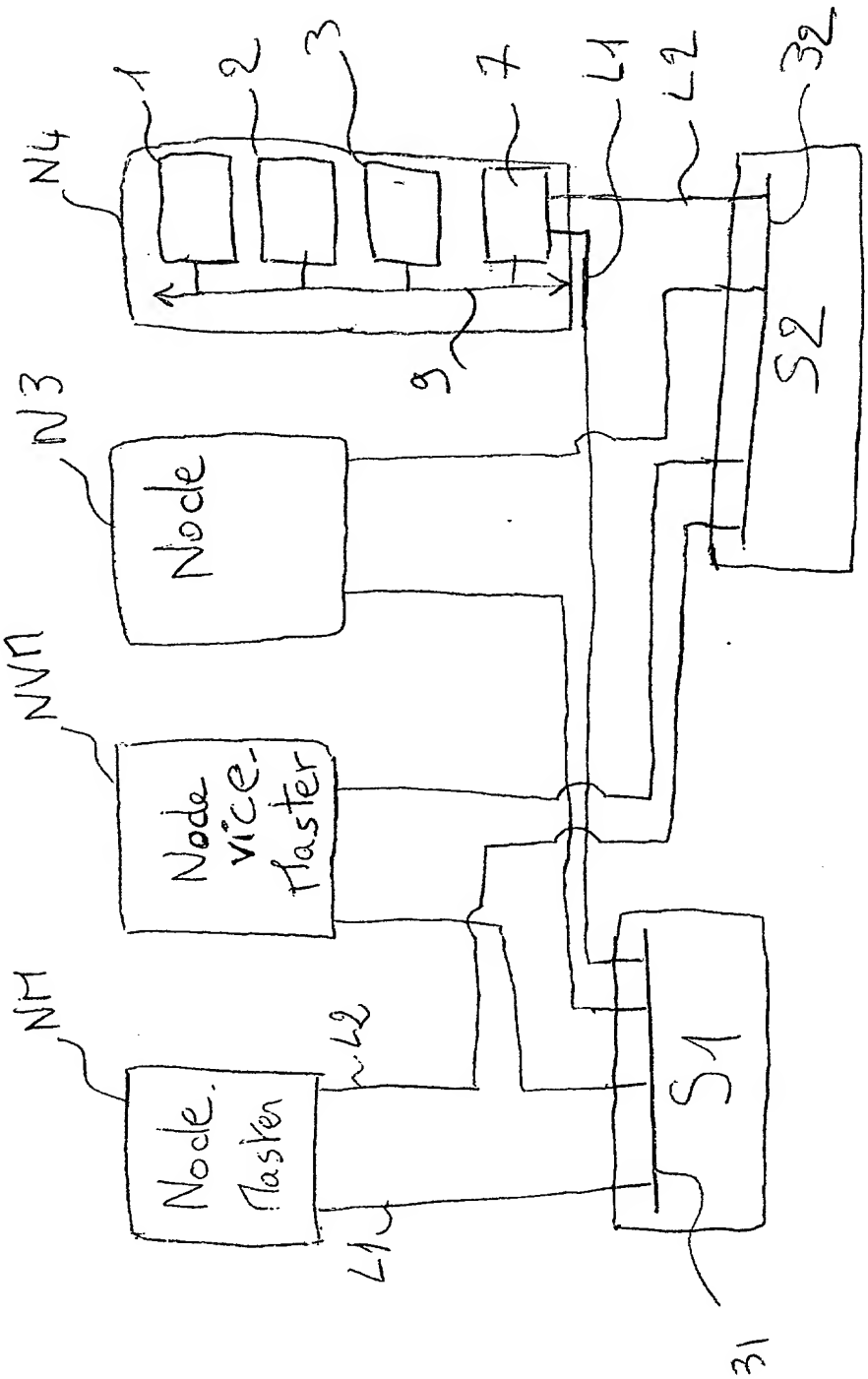


FIG.1

CABINET NETTER

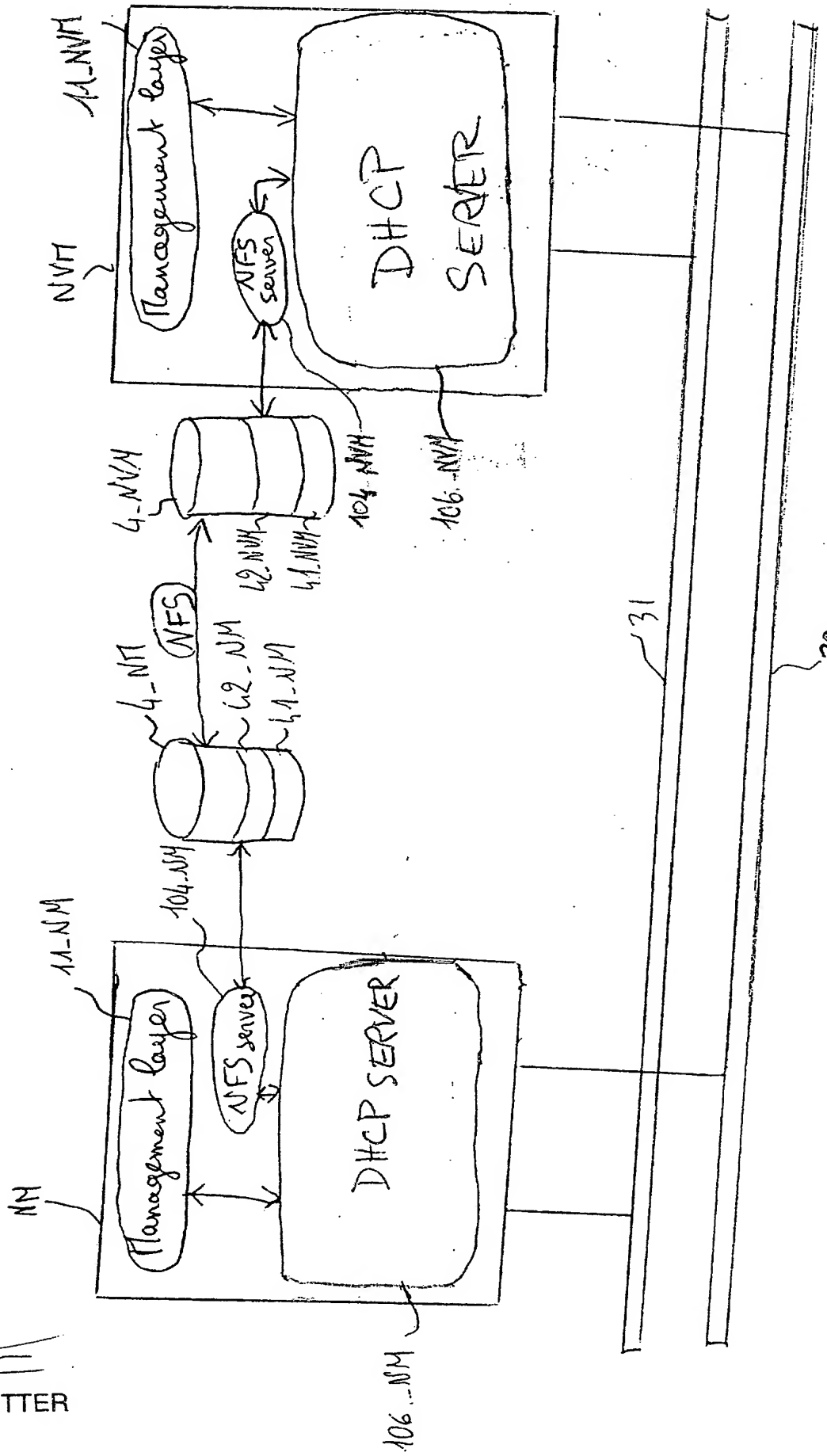


FIG 2

CABINET NETTER

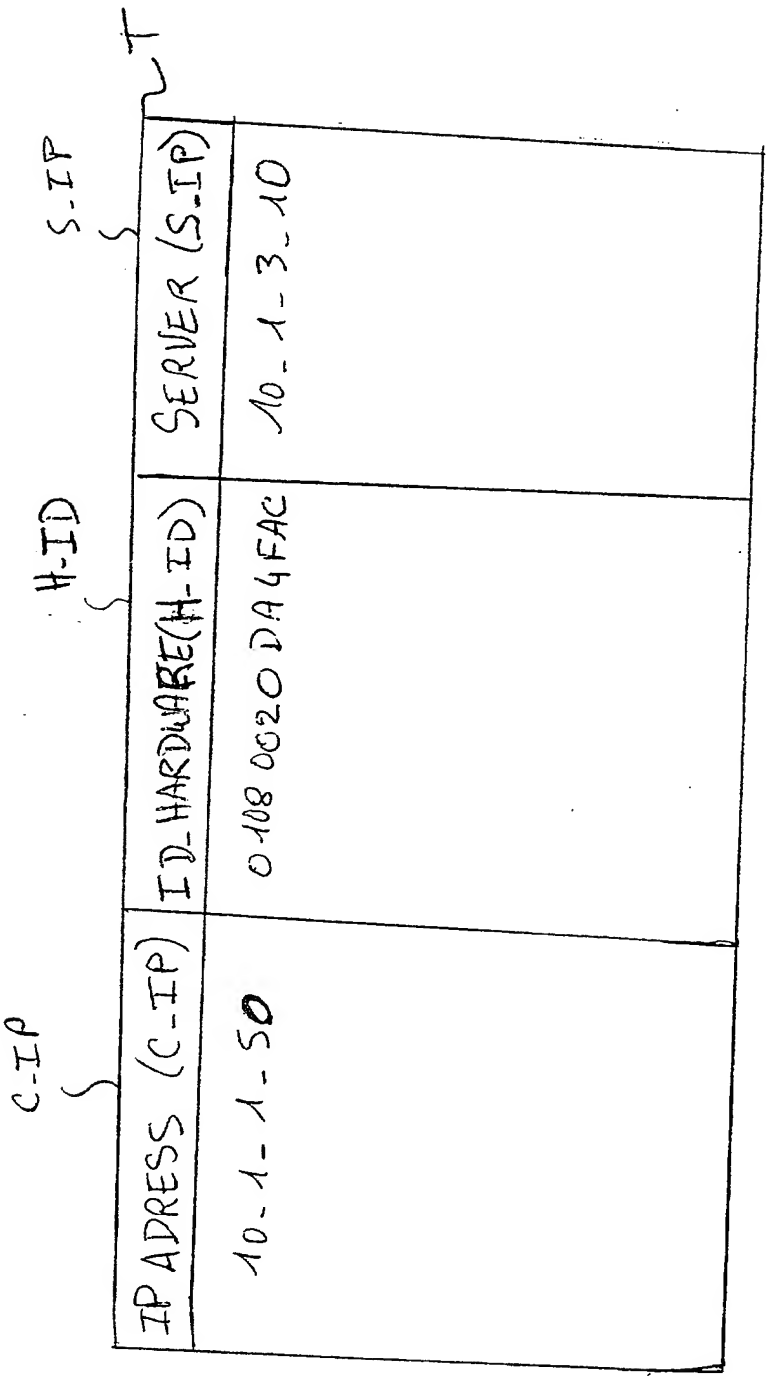


FIG-3

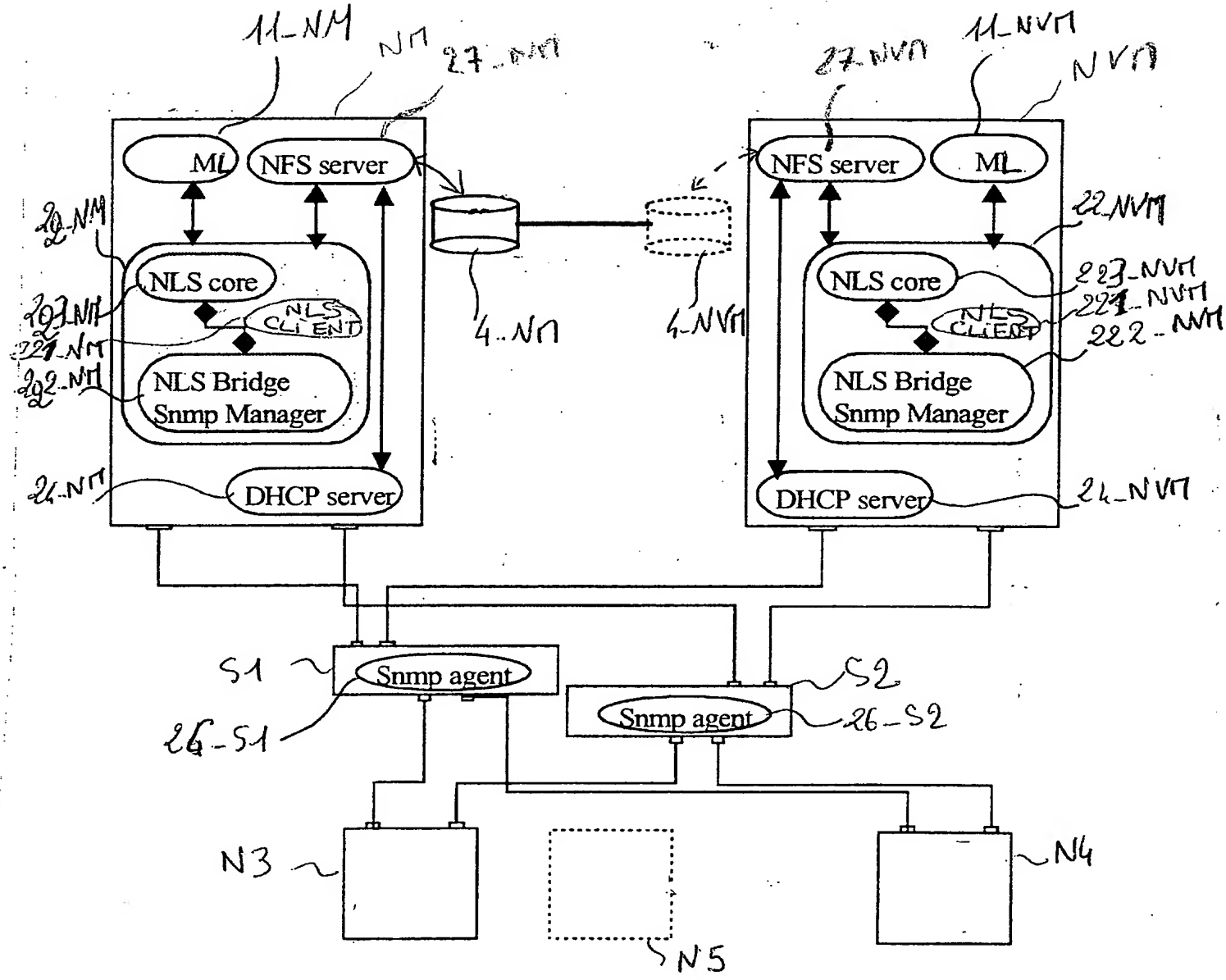


FIG 4

CABINET NETTER

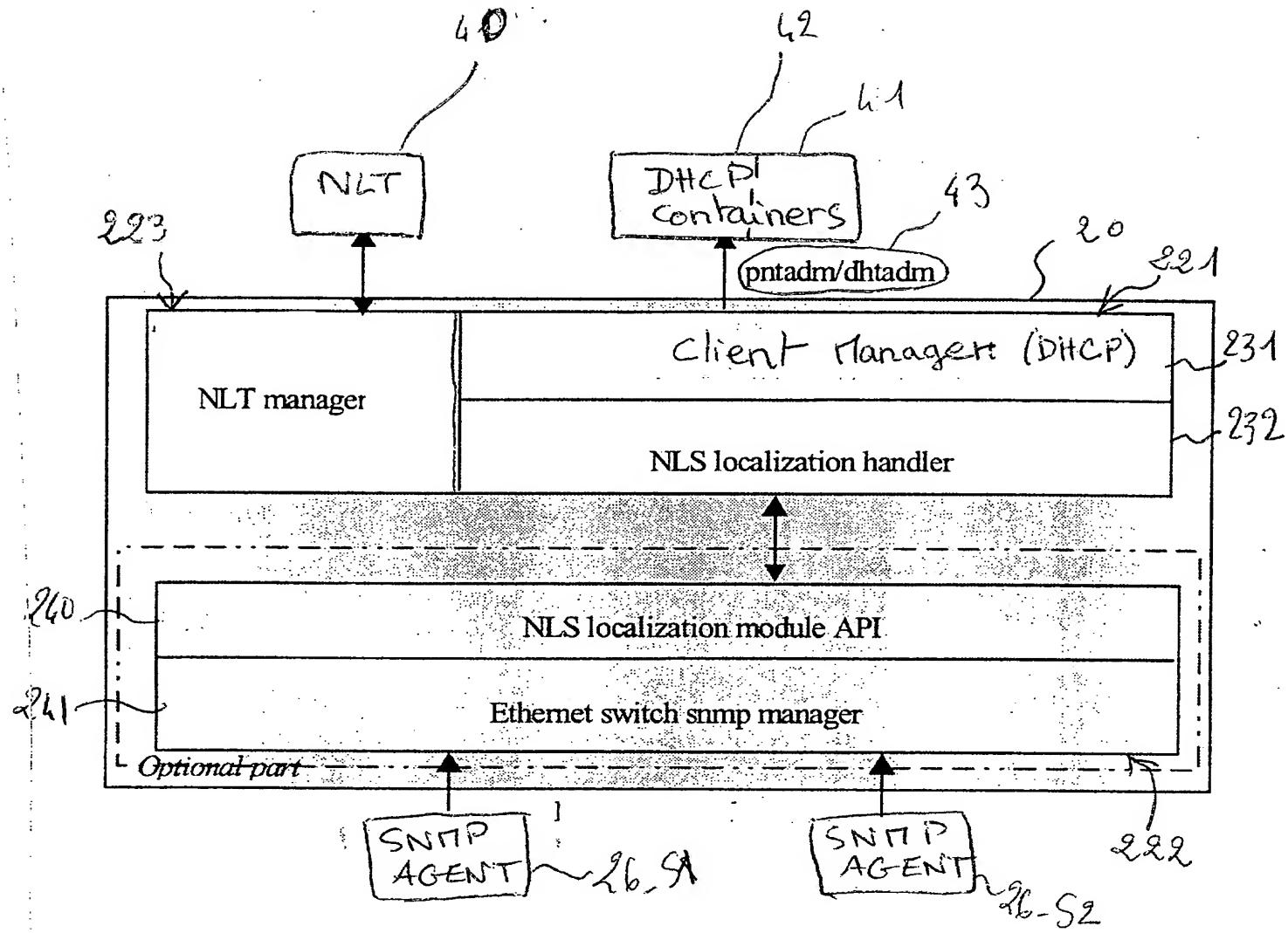


FIG 5

CABINET METTER

N_ID	Loc_T	Loc_ID	H_ID	BPar
21	01	10.11.1.0	0800020f99914	BootFile = inetboot. sun4u.Solaris_8

NLT

FIG 6

1er dépôt 6/12/18

CABINET METTER

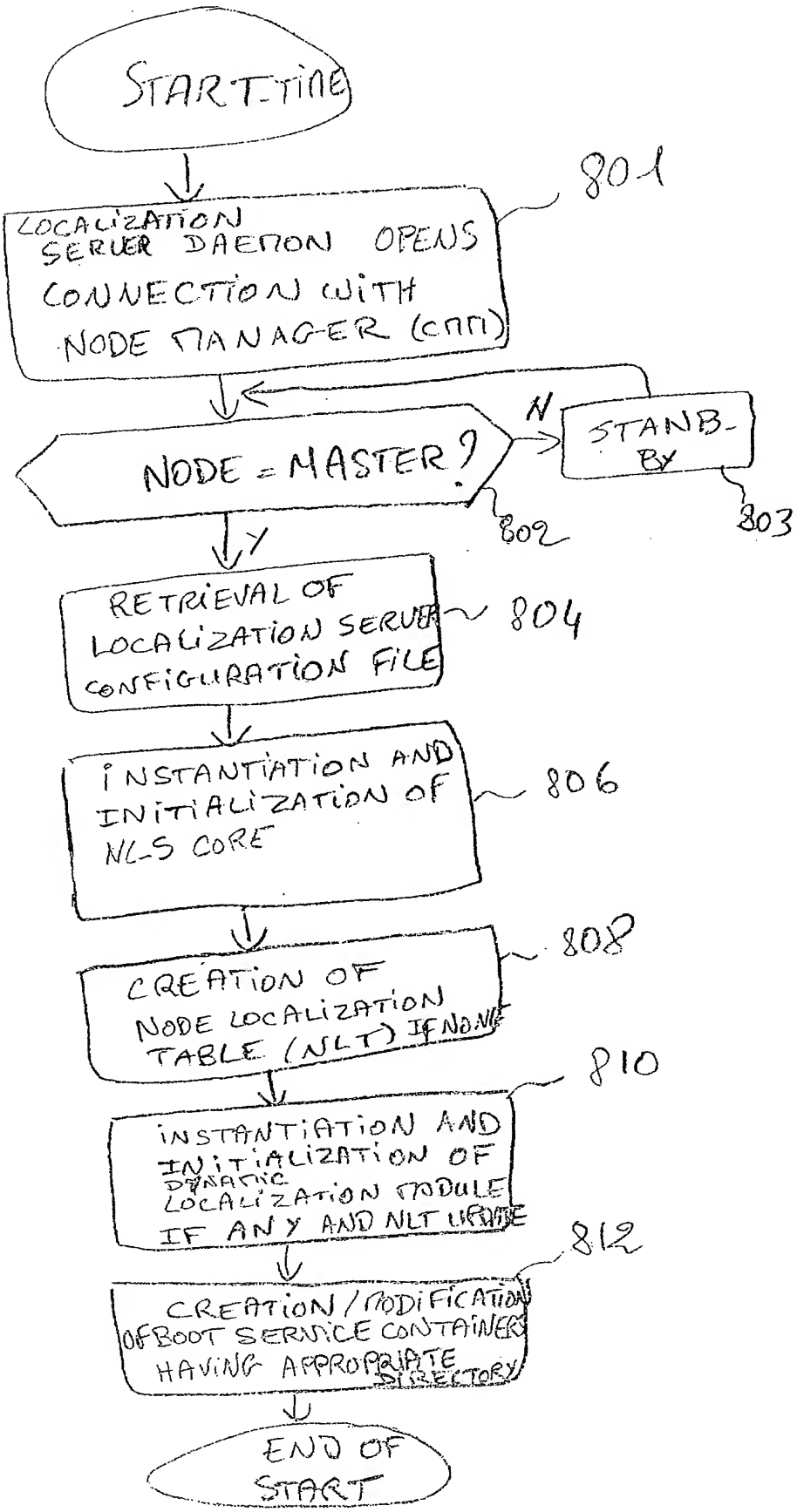


FIG 7

CABINET NETTER

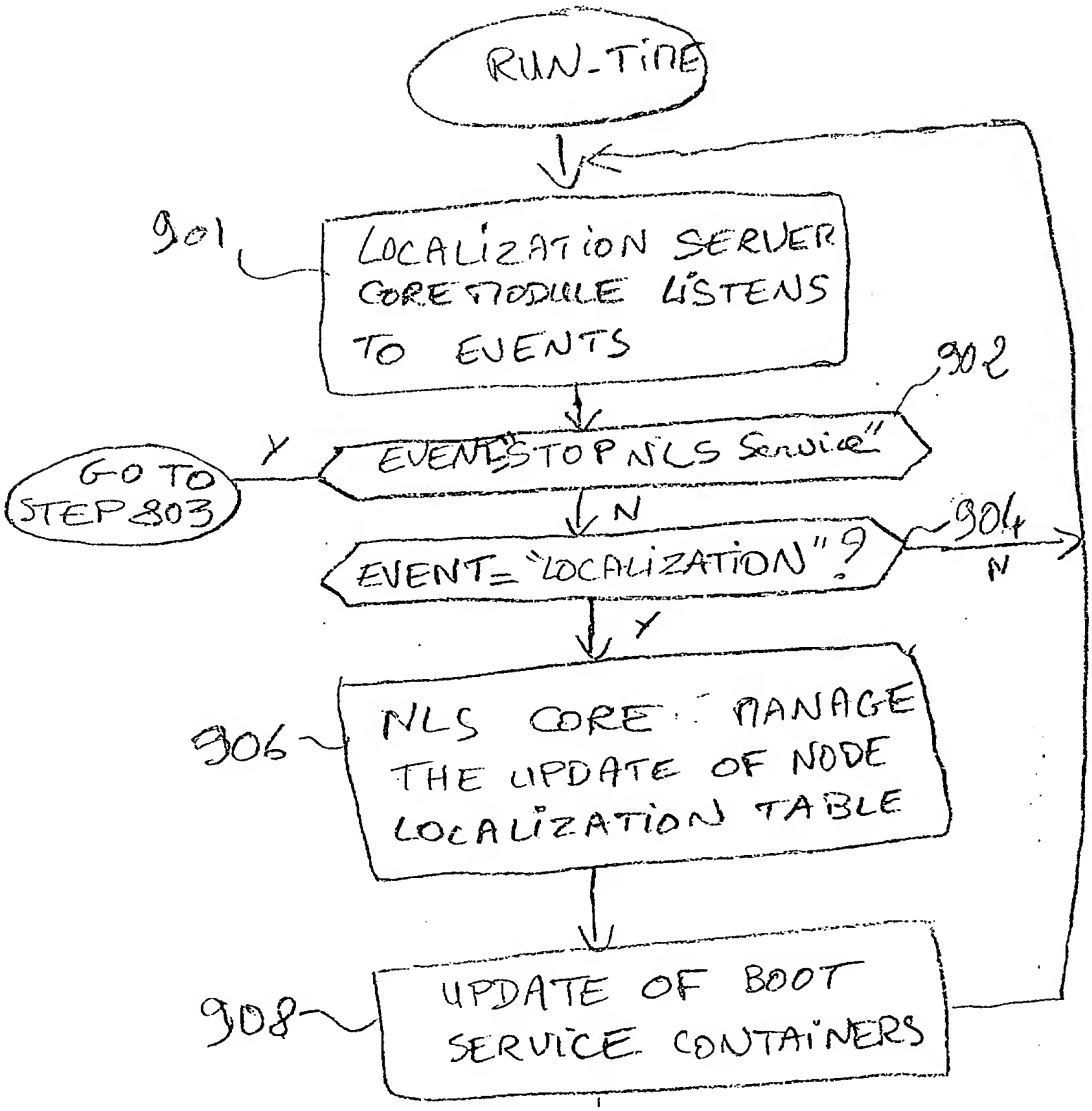
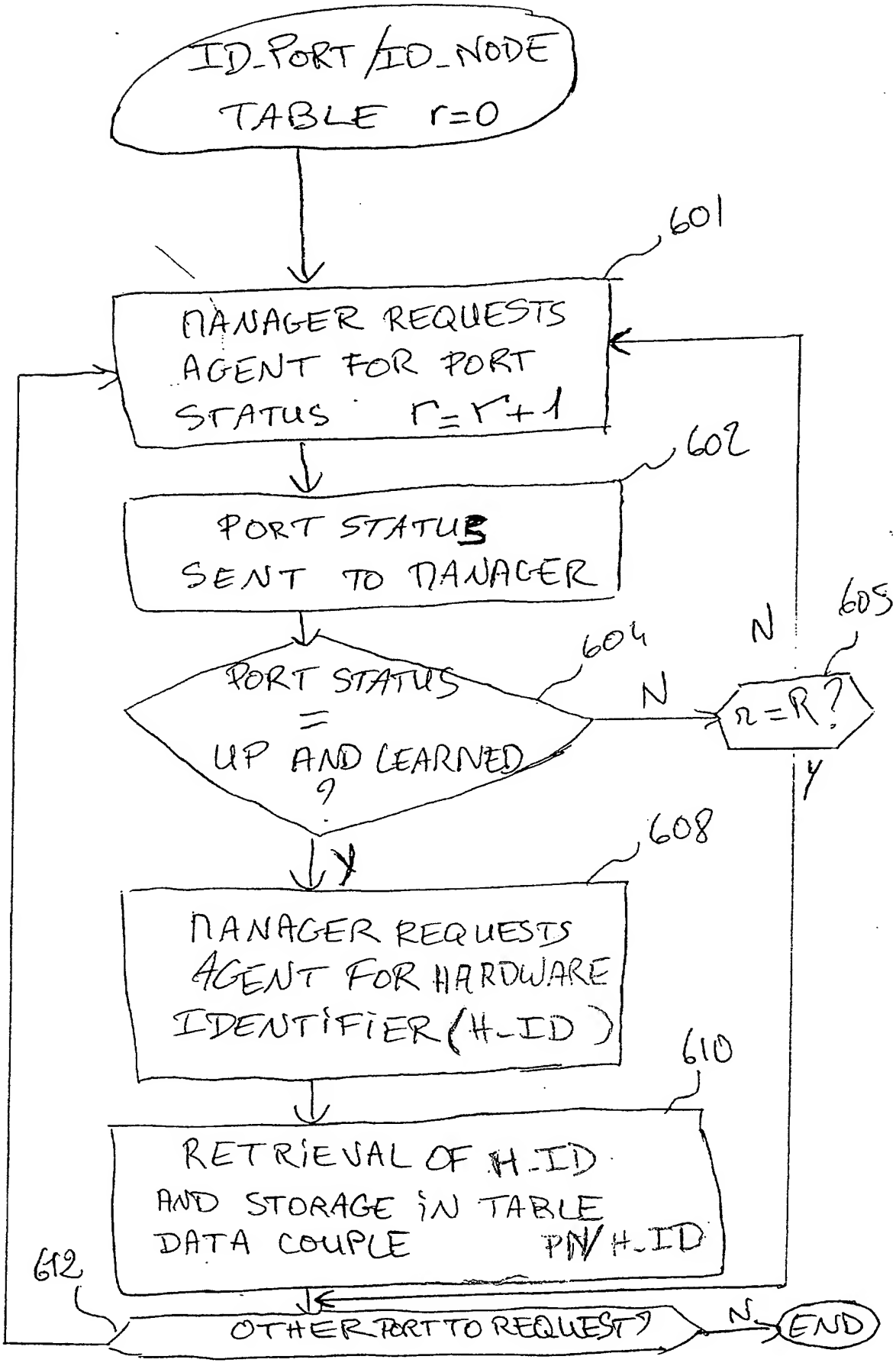

RUN-TIME
901
LOCALIZATION SERVER CORE MODULE LISTENS TO EVENTS
902
EVENT="STOP NLS Service"
Y
GO TO STEP 803
N
EVENT="LOCALIZATION"?
904
N
Y
906
NLS CORE : MANAGE THE UPDATE OF NODE LOCALIZATION TABLE
908
UPDATE OF BOOT SERVICE CONTAINERS

FIG 8

FIG 9



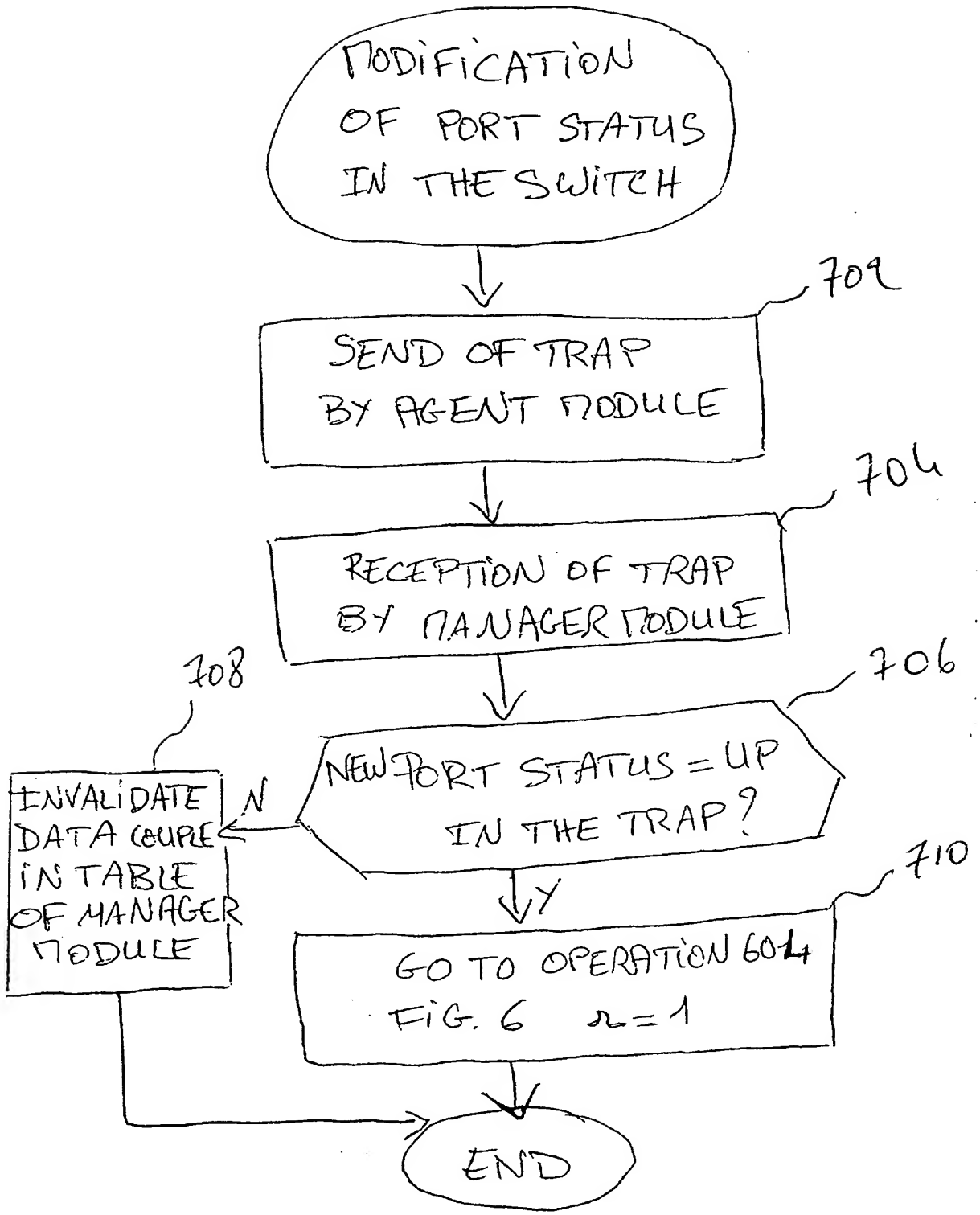


FIG 10 -

CABINET NETTER